KONFIGURASI JARINGAN

4

OBJEKTIF :

1. Mahasiswa Mengetahui Istilah-Istilah Yang Digunakan Pada Jaringan.
2. Mahasiswa Mengetahui Perintah-Perintah Yang Digunakan Untuk Mengonfigurasi Jaringan Pada Linux.

PENDAHULUAN

Memiliki akses ke jaringan adalah sebuah fitur utama dari kebanyakan sistem Linux. Pengguna dapat menggunakan internet seperti *browsing*, menerima dan mengirim email, dan mengirim *file* dengan pengguna lainnya.

Biasanya program untuk melakukan hal tersebut, seperti web *browser* dan email *client* cukup mudah digunakan. Bagaimanapun, itu semua bergantung pada sebuah fitur penting yang membuat komputer dapat berkomunikasi dengan komputer lainnya. Untuk dapat berkomunikasi, Anda harus mengetahui bagaimana untuk mengatur jaringan pada sistem.

Linux menyediakan beragam *tools* untuk mengatur dan mengonfigurasi jaringan serta untuk memantau kinerjanya.

4.1 ISTILAH-ISTILAH DASAR PADA JARINGAN

Sebelum mulai mengatur sebuah jaringan atau mengakses jaringan yang sudah ada, sangat penting untuk mengetahui beberapa istilah yang berkaitan erat dengan jaringan. Pada bagian ini dijelaskan beberapa istilah yang harusnya sudah familiar dengan Anda. Beberapa merupakan istilah dasar, dan mungkin Anda sudah mengetahuinya.

Host	*Host* adalah sebuah komputer. Mungkin kebanyakan orang langsung beranggapan tentang komputer PC atau laptop ketika mendengar istilah komputer. Tapi pada nyatanya, perangkat lain seperti handphone, music player, dan televisi juga merupakan komputer. Dalam lingkup jaringan, *host* adalah perangkat apapun yang berkomunikasi lewat jaringan dengan perangkat lain.
Network	*Network* atau jaringan adalah kumpulan dari dua atau lebih *host* (komputer) yang dapat berkomunikasi antara satu dengan lainnya. Komunikasi tersebut dapat lewat koneksi kabel atau nirkabel (*wireless*).
Internet	Internet adalah contoh dari jaringan. Pada internet terdiri dari jaringan yang dapat diakses *public* yang menghubungkan jutaan *host* dari seluruh dunia. Banyak orang menggunakan internet untuk menjelajahi *website* dan bertukar email, namun masih memiliki banyak kemampuan yang dapat dilakukan dengan internet selain hal tersebut.
Wi-Fi	Istilah Wi-Fi mengacu pada jaringan nirkabel (*wireless*)
Server	*Host* yang menyediakan layanan untuk *host* lain atau *client* disebut dengan *server*. Contohnya, web *server* untuk menyimpan, mengolah dan menyampaikan web *page*. Email *server* untuk menerima email yang masuk dan untuk mengirim email.
Service	Fitur yang disediakan oleh *host* disebut dengan *service*. Contohnya ketika sebuah *host* menyediakan web *page* untuk *host* lainnya.
Client	*Client* adalah *host* yang mengakses *server*. Ketika Anda menggunakan komputer untuk *browsing*, menonton film, dan lainnya, berarti Anda dapat disebut sebagai *client*.

Router	Disebut juga sebagai *gateway.* Router adalah mesin yang menghubungkan *host* dari satu jaringan ke jaringan lainnya. Contohnya, jika Anda bekerja di lingkungan kantor, semua komputer yang ada pada kantor Anda dapat terhubung satu dengan lainnya dengan *local network* yang dibuat oleh administrator. Untuk mengakses internet, komputer harus terhubung dengan router yang digunakan untuk meneruskan komunikasi jaringan ke internet. Biasanya, ketika Anda berkomunikasi di jaringan yang besar (seperti internet), beberapa router digunakan sebelum komunikasi Anda sampai pada tujuan akhir.

Selain istilah jaringan yang sudah dibahas sebelumnya, terdapat beberapa istilah lainnya yang harus Anda ketahui. Istilah-istilah berikut lebih fokus pada berbagai jenis layanan jaringan yang umum digunakan, serta beberapa teknik yang digunakan untuk berkomunikasi antar mesin.

Packet	*Network packet* digunakan untuk mengirim komunikasi jaringan antar *host*. Dengan memecah komunikasi tersebut ke dalam bagian yang lebih kecil (paket), sehingga pengiriman data jauh lebih efisien
IP Address	*Internet Protocol* (IP) *address* merupakan nomor unik yang ditetapkan pada *host* di jaringan. *Host* menggunakan nomor tersebut untuk menangani komunikasi jaringan
Mask	Disebut juga sebagai *netmask*, *subnet mask* atau *mask*. *Mask* merupakan nomor yang dapat digunakan untuk menentukan IP *address* mana yang dianggap berada dalam satu jaringan. Karena

	jika mana router menjalankan fungsinya, jaringan harus didefinisikan dengan jelas.
Hostname	Setiap *host* pada jaringan harus memiliki *hostname*-nya sendiri karena "nama" akan lebih mudah diingat oleh manusia dibanding dengan nomor, membuatnya lebih mudah untuk kita untuk mengalamatkan paket jaringan ke *host* lain. *Hostname* akan diubah menjadi alamat IP sebelum paket dikirim pada jaringan.
URL	*Uniform Resource Locator* (URL), sering disebut juga sebagai alamat web, digunakan untuk menentukan lokasi sumber, seperti halaman web (web *page*) pada internet. Inilah yang biasanya kita tuliskan pada *browser* untuk mengakses suatu web. Contohnya, http://www.netdevgroup.com. URL tersebut terdiri dari protocol yaitu http:// dan hostname www.netdevgroup.com
DHCP	*Host* dapat diberi *hostname*, alamat IP, dan informasi terkait jaringan lainnya oleh DHCP (*Dynamic Host Configuration Protocol*) *server*. Dalam dunia komputer, *protocol* adalah seperangkat aturan yang didefinisikan dengan baik. DHCP menentukan bagaimana informasi jaringan ditetapkan ke *host client*, dan DHCP *server* ini merupakan mesin yang menyediakan informasi tersebut.
DNS	Seperti yang sudah disebutkan sebelumnya, *hostname* diubah mejadi alamat IP sebelum paket dikirim pada jaringan. Jadi sebuah *host* harus tahu alamat IP dari semua *host* lain yang berkomunikasi dengan *host* tersebut. Ketika bekerja pada jaringan yang besar (seperti internet), ini akan menjadi tantangan karena ada begitu banyak *host*. DNS (*Domain Name System*) berfungsi untuk mengubah nama domain ke alamat IP.

Ethernet	Dalam lingkungan jaringan berkabel (*wired network*), menggunkan *ethernet* adalah cara paling umum untuk mengubungkan *host* ke jaringan secara fisik. Kabel *ethernet* terhubung dengan kartu jaringan (*network card*) yang mendukung koneksi *ethernet*. Kabel *ethernet* dan perangkat jaringan (seperti router) dirancang khusus untuk mendukung kecepatan komunikasi yang berbeda. Kecepatan terendah adalah 10 Mbps dan yang paling tinggi adalah 100 Gbps. Kecepatan dengan *ethernet* yang paling umum adalah 100 Mbps dan 1 Gbps.
TCP/IP	*Transmission Control Protocol/Internet Protocol* (TCP/IP) adalah sebutan untuk kumpulan *protocol* (ingat, *protocol* adalah seperangkat aturan) yang digunakan untuk menentukan bagaimana komunikasi harus dilakukan antar *host*. Meskipun ini bukan satu-satunya kumpulan *protocol* yang digunakan untuk menentukan komunikasi jaringan, tapi TCP/IP merupakan yang paling sering digunakan. Sebagai contoh, TCP/IP mencakup tentang cara kerja alamat IP dan *network mask*.

ALAMAT IP

Seperti yang sudah disebutkan sebelumnya, *host* menyampaikan paket dengan menggunakan alamat IP sebagai tujuannya. Paket jaringan juga mencakup *return address*, yang merupakan alamat IP mesin pengirim. Terdapat dua jenis alamat IP: **IPv4** dan **IPv6**. Untuk memahami mengapa ada dua jenis alamat IP yang berbeda, Anda harus mengetahui dulu sedikit mengenai sejarah pengalamatan IP.

Selama bertahun-tahun, Teknik pengalamatan IP yang digunakan oleh semua komputer adalah Ipv4. Pada IPv4, terdapat total 4 angka 8-bit yang digunakan untuk menentukan alamat. Ini dikenal sebagai alamat 32-bit (4 x 8 = 32). Contohnya:

```
192.168.10.120.
```

8-bit mengacu pada angka dari 0 hingga 255.

Setiap *host* pada internet harus memiliki alamat IP yang berbeda. Pada IPv4, terdapat batas penggunaan alamat IP yaitu sekitar 4,3 milyar alamat IP. Namun, banyak dari alamat IP tersebut yang tidak dapat digunakan karena berbagai alasan. Dan juga, banyak organisasi belum menggunakan semua alamat IP yang tersedia.

Meskipun tampaknya ada banyak alamat IP yang dapat digunakan, berbagai faktor telah menyebabkan satu masalah yaitu: internet mulai kehabisan alamat IP. Masalah tersebut mendorong pengembangan IPv6. IPv6 secara resmi dibuat pada 1998. Dalam jaringan IPv6, alamatnya jauh lebih besar, dengan alamat 128-bit yang terlihat seperti berikut:

```
2001:0db8:85a3:0042:1000:8a2e:0370:7334
```

Pada dasarnya, IPv6 menyediakan kumpulan alamat yang jauh lebih besar. Saking besarnya sehingga hampir tidak mungkin kehabisan alamat IP dalam waktu dekat. Penting untuk diketahui, perbedaan antara IPv4 dan IPv6 bukan hanya sekedar besarnya. IPv6 memiliki berbagai kelebihan yang masih terbatas di IPv4, seperti kecepatan jaringan yang lebih cepat, manajemen paket yang lebih baik, dan transportasi/pengiriman data yang lebih efisien.

Menyadari kelebihan tersebut, pasti Anda berpikir mulai sekarang semua *host* akan menggunakan IPv6. Namun, mayoritas perangkat yang terhubung ke jaringan di dunia saat ini masih menggunakan IPv4 (sekitar 98-99% dari semua perangkat).

Jadi, mengapa di dunia ini belum memaksimalkan penggunaan teknologi IPv6 yang lebih canggih? Terdapat dua alasan utama:

- **NAT**: NAT (*Net Address Translation*) diciptakan untuk mengatasi kemungkinan kehabisan alamat IP pada IPv4. NAT menggunakan teknik

untuk menyediakan lebih banyak *host* yang terhubung ke internet. Singkatnya, sekelompok *host* ditempatkan pada jaringan pribadi (*private network*) tanpa akses langsung ke internet; router khusus menyediakan akses internet, dan hanya router ini yang membutuhkan alamat IP untuk dapat berkomunikasi di internet. Dengan kata lain, sekumpulan *host* berbagi satu alamat IP, yang berarti lebih banyak komputer yang dapat terhubung ke internet. Fitur ini berarti kebutuhan untuk pindah ke IPv6 kurang penting dibandingkan sebelum ditemukannya NAT.

- **Porting**: *Porting* berpindah dari satu teknologi ke teknologi lainnya. IPv6 memiliki banyak fitur baru yang hebat, tapi semua *host* harus dapat memanfaatkan fitur-fituur tersebut. Membuat semua orang di internet (atau bahkan beberapa) untuk membuat perubahan ini merupakan suatu tantangan tersendiri.

Meskipun demikian, Sebagian besar ahli setuju bahwa IPv6 akan menggantikan IPv4, jadi memahami dasar-dasar keduanya sangat disarankan bagi mereka yang bekerja di bidang industri IT.

4.2 KONFIGURASI JARINGAN PADA LINUX

Ketika Anda mengkonfigurasi perangkat jaringan, ada dua hal awal yang perlu Anda pertimbangkan:

- **Berkabel (*wired*) atau nirkabel (*wireless*)?** Konfigurasi perangkat nirkabel sedikit berbeda dengan konfigurasi perangkat berkabel karena beberapa fitur tambahan yang biasanya ditemukan pada perangkat nirkabel (seperti keamanan).
- **DHCP atau alamat statis (*static address*)?** Ingatlah bahwa server DHCP menyediakan informasi jaringan, seperti alamat IP dan *subnet mask*. Jika Anda tidak menggunakan server DHCP, maka Anda harus memberikan

informasi secara manual ke *host* Anda, yang disebut menggunakan alamat IP statis. Secara umum, mesin desktop menggunakan jaringan kabel, sedangkan laptop menggunakan jaringan nirkabel. Biasanya mesin berkabel menggunakan alamat IP statis, tetapi ini juga sering ditetapkan melalui server DHCP. Di hampir semua kasus, mesin nirkabel menggunakan DHCP karena mereka hampir selalu bergerak dan terhubung ke jaringan yang berbeda.

Secara umum, perangkat desktop menggunakan jaringan kabel, sedangkan laptop menggunakan jaringan nirkabel. Biasanya mesin berkabel menggunakan alamat IP statis, tetapi ini juga sering ditetapkan melalui server DHCP. Di hampir semua kasus, perangkat nirkabel menggunakan DHCP karena mereka hampir selalu bergerak dan terhubung ke jaringan yang berbeda.

MENGONFIGURASI JARINGAN DENGAN MENGGUNAKAN FILE KONFIGURASI

File konfigurasi digunakan untuk menyimpan dan memodifikasi data jaringan. *File* konfigurasi utama untuk jaringan IPv4 adalah *file* `/etc/sysconfig/network-scripts/ifcfg-eth0`. Berikut ini contoh tampilan *file* ketika dikonfigurasi untuk alamat IP statis:

```
root@localhost:~# cat /etc/sysconfig/network-scripts/ifcfg-eth0
DEVICE="eth0"
BOOTPROTO=none
NM_CONTROLLED="yes"
ONBOOT=yes
TYPE="Ethernet"
UUID="98cf38bf-d91c-49b3-bb1b-f48ae7f2d3b5"
DEFROUTE=yes
IPV4 _FAILURE_FATAL=yes
IPV6INOT=no
```

```
NAME="System eth0"
IPADDR=192.168.1.1
PREFIX=24
GATEWAY=192.168.1.1
DNS1=192.168.1.2
HWADDR=00:50:56:90:18:18
LAST_CONNECT=1376319928
```

Jika perangkat dikonfigurasi dengan DHCP, nilai BOOTPROTO akan diubah menjadi dhcp, dan nilai IPADDR, GATEWAY dan DNS1 dikosongkan.

FILE KONFIGURASI PADA IPv6

Pada sistem CentOS, *file* konfigurasi utama jaringan IPv6 adalah *file* yang sama dengan *file* konfigurasi IPv4 disimpan; yaitu di `/etc/sysconfig/network-scripts/ifcfg-eth0`. Jika Anda ingin sistem Anda memiliki alamat IPv6 statis, tambahkan berikut ini ke *file* konfigurasi:

```
IPV6INIT=yes
IPV6ADDR=<IPv6 IP Address>
IPV6_DEFAULTGW=<IPv6 IP Gateway Address>
```

Jika Anda ingin sistem Anda menjadi klien DHCP IPv6, tambahkan pengaturan berikut:

```
DHCPV6C=yes
```

Anda juga harus menambahkan pengaturan berikut pada file `/etc/sysconfig/network`:

```
NETWORKING_IPV6=yes
```

Perlu diingat

Metode yang diterima secara luas untuk membuat perubahan pada antarmuka jaringan adalah dengan menghapus antarmuka menggunakan perintah seperti ifdown eth0, membuat perubahan yang diinginkan pada *file* konfigurasi, dan kemudian mengembalikan antarmuka ke layanan dengan perintah ifup eth0.

Metode lain yang kurang spesifik adalah dengan memulai ulang jaringan sistem sepenuhnya, dengan perintah seperti *service network restart*, yang menghapus semua antarmuka, membaca ulang semua *file* konfigurasi terkait, dan kemudian memulai ulang jaringan untuk sistem.

Memulai ulang layanan jaringan dapat mengganggu lebih dari sekedar antarmuka tunggal yang ingin diubah pengguna, jadi gunakan perintah yang paling terbatas dan spesifik untuk memulai ulang antarmuka jika memungkinkan.

Contoh berikut menunjukkan bagaimana perintah *service* perlu dijalankan pada sistem CentOS:

```
[root@localhost ~]# service network restart
Shutting down interface eth0:  Device state: 3 (disconnected)
                                                           [  OK  ]
Shutting down loopback interface:                          [  OK  ]
Bringing up loopback interface:                            [  OK  ]
Bringing up interface eth0:  Active connection state: activated
Active connection path: /org/freedesktop/NetworkManager/ActiveConnection/
1
                                                           [  OK  ]
```

DOMAIN NAME SYSTEM (DNS)

Saat komputer diminta untuk mengakses situs web, seperti www.example.com, komputer tidak selalu mengetahui alamat IP yang akan digunakan. Untuk komputer agar mengasosiasikan alamat IP dengan URL atau permintaan nama *host*, komputer bergantung pada layanan DNS dari komputer lain. Seringkali, alamat IP dari *server* DNS ditemukan ketika sedang melakukan

permintaan DHCP, saat komputer menerima informasi pengalamatan penting untuk berkomunikasi di jaringan.

Alamat DNS *server* disimpan di *file* `/etc/resolv.conf`. *File* `/etc/resolv.conf` dibuat secara otomatis dan terlihat seperti berikut:

```
sysadmin@localhost:~$ cat /etc/resolv.conf
nameserver 127.0.0.1
```

Pengaturan *name server* sering kali diatur ke alamat IP dari DNS *server*. Contoh berikut menggunakan perintah `host`, yang bekerja dengan DNS untuk mengasosiasikan nama *host* dengan alamat IP. Perhatikan bahwa *server* contoh dikaitkan dengan alamat IP 192.168.1.2 oleh DNS *server*:

```
sysadmin@localhost:~$ host example.com
example.com has address 192.168.1.2
```

Ini juga umum untuk memiliki beberapa pengaturan *name server*, jika satu server DNS tidak merespons.

FILE KONFIGURASI JARINGAN

Name resolution pada *host* Linux dilakukan dengan 3 *file* penting: *file* `/etc/hosts`, `/etc/resolv.conf` dan `/etc/nsswitch.conf`. Bersama-sama, mereka menjelaskan lokasi informasi layanan nama, urutan untuk memeriksa sumber daya, dan ke mana harus mencari informasi itu.

File	Penjelasan
`/etc/hosts`	*File* ini berisi tabel nama *host* ke alamat IP yang dapat digunakan untuk melengkapi DNS *server*.

```
sysadmin@localhost:~$ cat /etc/hosts
127.0.0.1       localhost
```

`/etc/resolv.conf`

File ini berisi alamat IP dari nama *server* yang harus diperiksa oleh sistem dalam upaya untuk mengubah nama ke alamat IP. *Server-server* ini seringkali adalah DNS *server*. I ni juga dapat berisi kata kunci dan nilai tambahan yang dapat mempengaruhi proses resolusi.

```
sysadmin@localhost:~$ cat /etc/resolv.conf
nameserver 127.0.0.11
```

`/etc/nsswitch.conf`

File ini dapat digunakan untuk mengubah tempat pencarian nama *host*. Berisi entri tertentu yang menjelaskan dalam urutan apa sumber *name resolution* diperiksa.

```
sysadmin@localhost:~$ cat /etc/nsswitch.conf
# /etc/nsswitch.conf
#

Output Omitted...

hosts:          files dns

Output Omitted...
```

File `/etc/hosts` dicari pertama. DNS *server* setelahnya:

```
hosts: files dns
```

DNS *server* dicari pertama, baru *local file* dicari setelahnya:

```
hosts: dns files
```

Perintah atau program pada sistem, seperti *browser*, akan terhubung dengan komputer lain dengan DNS *name*. Kemudian sistem akan mengcek berbagai *file* dalam urutan tertentu untuk mengubah *hostname* tersebut menjadi alamat IP yang dapat digunakan.

1. Pertama akan mengecek *file* `/etc/nsswitch.conf`

```
hosts:          files dns
```

   Baris tersebut menunjukkan bahwa sistem harus mengecek *local file* terlebih dahulu untuk mengubah *hostname*, yang berarti *file* `/etc/hosts` akan diurai agar cocok dengan nama yang diminta.

2. Kedua, sistem akan mengecek *file* `/etc/hosts` untuk mencoba menentukan nama. Jika namanya cocok dengan entri yang ada di `/etc/hosts`, maka akan langsung diubah.

   JIka hal tersebut terjadi, maka tidak akan melakukan *failover* (atau melanjutkan) ke opsi DNS, meskipun perubahan tersebut tidak akurat. Ini dapat terjadi jika entri di `/etc/hosts` mengarah ke alamat IP yang tidak ditetapkan.

3. Ketiga, jika *file* lokal `/etc/hosts` tidak menghasilkan kecocokan, sistem akan menggunakan entri pada DNS *server* yang dikonfigurasi yang terdapat dalam *file* `/etc/resolv.conf` untuk mencoba menemukan nama tersebut.

   FIle `/etc/resolv.conf` harus berisi setidaknya dua entri nama *server*, seperti *file* contoh di bawah ini:

```
nameserver 10.0.2.3
nameserver 10.0.2.4
```

Sistem resolusi DNS akan menggunakan nama *server* pertama untuk percobaan pencarian nama. Jika itu tidak tersedia, atau periode waktu habis tercapai, server kedua kemudian akan ditanyai untuk resolusi nama. Jika

kecocokan ditemukan, maka akan dikembalikan ke sistem dan digunakan untuk memulai koneksi dan juga ditempatkan di DNS *cache* untuk jangka waktu yang dapat diatur.

Perlu diingat

Dua istilah lain yang mungkin muncul pada file /etc/resolv.conf. Meskipun ini berada di luar cakupan kursus ini, mereka secara rutin disertakan dalam file default /etc/resolv.conf sehingga kami menyertakan penjelasan dari istilah-istilah berikut di bawah ini:

`domain` Diikuti oleh *domain* yang memenuhi syarat, seperti snowblower.example.com, memungkinkan kueri untuk polaris *host* untuk dicoba sama seperti polaris *host*, atau jika gagal, menambahkan sisa nama *domain* ke dalamnya dan semoga diselesaikan dengan *server* sebagai nama itu (misalnya polaris.snowblower.example.com.).

`search` Diikuti oleh sekumpulan *domain* terpisah yang dapat ditanyai satu demi satu semoga bisa menemukan namanya.

4.3 TOOLS JARINGAN

Ada beberapa perintah yang dapat Anda gunakan untuk melihat informasi jaringan. *Tools* berikut ini juga dapat berguna saat Anda memecahkan masalah jaringan.

PERINTAH ifconfig

Perintah ifconfig adalah singkatan dari "*interface configuration*" dan digunakan untuk menampilkan informasi konfigurasi jaringan. Tidak semua pengaturan jaringan tercakup dalam materi ini, tetapi penting untuk diperhatikan dari *output* di bawah ini adalah alamat IP perangkat jaringan utama eth0 adalah 192.168.1.2 dan perangkat saat ini aktif (UP):

```
root@localhost:~# ifconfig
```

```
eth0      Link encap:Ethernet  HWaddr b6:84:ab:e9:8f:0a
          inet addr:192.168.1.2  Bcast:0.0.0.0  Mask:255.255.255.0
          inet6 addr: fe80::b484:abff:fee9:8f0a/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:95 errors:0 dropped:4 overruns:0 frame:0
          TX packets:9 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:25306 (25.3 KB)  TX bytes:690 (690.0 B)
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:6 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:460 (460.0 B)  TX bytes:460 (460.0 B)
```

Perangkat lo disebut sebagai perangkat *loopback*. Ini adalah perangkat jaringan khusus yang digunakan oleh sistem saat mengirim data berbasis jaringan ke dirinya sendiri.

Perintah ifconfig juga dapat digunakan untuk mengubah pengaturan jaringan sementara. Biasanya perubahan ini harus permanen, jadi menggunakan perintah ifconfig untuk membuat perubahan seperti itu relatif jarang.

PERINTAH ip

Perintah ifconfig menjadi usang di beberapa distro Linux (tidak digunakan lagi) dan diganti dengan perintah ip, khususnya ip addr show.

Perintah ip berbeda dari ifconfig dalam beberapa kondisi, terutama melalui peningkatan fungsionalitas dan serangkaian pengaturan, ip hampir bisa menjadi tempat serba ada untuk konfigurasi dan kontrol jaringan sistem. Format untuk perintah ip adalah sebagai berikut:

```
ip [OPTIONS] OBJECT COMMAND
```

Sementara ifconfig terbatas terutama dalam hal modifikasi parameter jaringan, dan menampilkan *detail* konfigurasi komponen jaringan, perintah ip dapat melakukan beberapa pekerjaan dari beberapa perintah lama lainnya seperti route dan arp.

Catatan: Perintah Linux dan Unix biasanya tidak hilang begitu saja saat menjadi usang; mereka tetap bertahan sebagai perintah lama, terkadang selama bertahun-tahun, karena jumlah skrip yang bergantung pada perintah tersebut, dan jumlah memori di antara administrator sistem, menjadikannya ide yang baik untuk menyimpannya demi kompatibilitas.

Perintah ip awalnya dapat terlihat sedikit lebih panjang daripada perintah ifconfig, tetapi ini adalah masalah penyusunan kata dan hasil dari filosofi di balik pengoperasian perintah ip.

Pada contoh di bawah ini, baik perintah ifconfig dan perintah ip digunakan untuk menampilkan semua antarmuka pada sistem.

```
root@localhost:~# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0c:29:71:f0:bb
          inet addr:172.16.241.140  Bcast:172.16.241.255  Mask:25
5.255.255.0
          inet6 addr: fe80::20c:29ff:fe71:f0bb/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:8506 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1201 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:8933700 (8.9 MB)  TX bytes:117237 (117.2 KB)


lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:285 errors:0 dropped:0 overruns:0 frame:0
          TX packets:285 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:21413 (21.4 KB)  TX bytes:21413 (21.4 KB)
root@localhost:~# ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOW
N group default qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
```

```
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fa
st state UP group default qlen 1000
    link/ether 00:0c:29:71:f0:bb brd ff:ff:ff:ff:ff:ff
    inet 172.16.241.140/24 brd 172.16.241.255 scope global eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe71:f0bb/64 scope link
       valid_lft forever preferred_lft forever
```

Keduanya menunjukkan jenis antarmuka, protokol, perangkat keras dan alamat IP, pelindung jaringan, dan berbagai informasi lainnya tentang masing-masing antarmuka yang aktif pada sistem.

PERINTAH route

Ingatlah bahwa router (atau gateway) adalah mesin yang memungkinkan *host* dari satu jaringan untuk berkomunikasi dengan jaringan lain. Untuk melihat tabel yang menjelaskan ke mana paket jaringan dikirim, digunakan perintah route:

```
root@localhost:~# route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     *               255.255.255.0   U     0      0        0 eth0
default         192.168.1.1     0.0.0.0         UG    0      0        0 eth0
```

Baris pertama yang disorot dalam contoh sebelumnya menunjukkan bahwa setiap paket jaringan yang dikirim ke mesin pada jaringan 192.168.1 tidak dikirim ke *gateway* (tanda * menunjukkan tidak ada *gateway*). Baris kedua yang disorot menunjukkan bahwa semua paket jaringan lainnya dikirim ke host dengan alamat IP 192.168.1.1 (router).

Beberapa pengguna lebih suka menampilkan informasi ini hanya dengan data numerik, dengan menggunakan opsi -n pada perintah rute. Misalnya, lihat

yang berikut ini dan fokus pada *output* yang digunakan untuk menampilkan *default*:

```
root@localhost:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     0.0.0.0         255.255.255.0   U     0      0        0 eth0
0.0.0.0         192.168.1.1     0.0.0.0         UG    0      0        0 eth0
```

0.0.0.0 merujuk ke semua mesin lain, dan sama dengan *default*.

Perintah route menjadi usang di beberapa distribusi Linux (tidak digunakan lagi) dan digantikan dengan perintah ip, khususnya ip route show. Perhatikan bahwa informasi yang sama yang disorot di atas juga dapat ditemukan menggunakan perintah ini:

```
root@localhost:~# ip route show
default via 192.168.1.254 dev eth0 proto static
192.168.1.0/24 dev eth0  proto kernel  scope link  src 192.168.1.2
```

PERINTAH ping

Perintah ping dapat digunakan untuk menentukan apakah mesin lain dapat dijangkau. Jika perintah ping dapat mengirim paket jaringan ke mesin lain dan menerima respons, maka Anda dapat terhubung ke mesin itu. Secara *default*, perintah ping terus mengirim paket tanpa henti. Untuk membatasi berapa banyak ping yang akan dikirim, gunakan opsi -c diikuti dengan angka yang menunjukkan berapa banyak iterasi yang Anda inginkan. Contoh berikut menunjukkan ping dibatasi hingga 4 iterasi. Jika perintah ping berhasil, akan terlihat seperti contoh berikut:

```
root@localhost:~# ping -c 4 192.168.1.2
PING 192.168.1.2 (192.168.1.2) 56(84) bytes of data.
64 bytes from 192.168.1.2: icmp_req=1 ttl=64 time=0.051 ms
64 bytes from 192.168.1.2: icmp_req=2 ttl=64 time=0.064 ms
64 bytes from 192.168.1.2: icmp_req=3 ttl=64 time=0.050 ms
```

```
64 bytes from 192.168.1.2: icmp_req=4 ttl=64 time=0.043 ms


--- 192.168.1.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 2999ms
rtt min/avg/max/mdev = 0.043/0.052/0.064/0.007 ms
```

Jika perintah ping gagal, maka akan muncul pesan "*Destination Host Unreachable*":

```
root@localhost:~# ping -c 4 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.
From 192.168.1.2 icmp_seq=1 Destination Host Unreachable
From 192.168.1.2 icmp_seq=2 Destination Host Unreachable
From 192.168.1.2 icmp_seq=3 Destination Host Unreachable
From 192.168.1.2 icmp_seq=4 Destination Host Unreachable


--- 192.168.1.1 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, ti
me 2999ms
pipe 4
```

Penting untuk dicatat apabila perintah ping gagal tidak berarti sistem jarak jauh tidak dapat dijangkau. Beberapa administrator mengkonfigurasi mesin mereka (dan bahkan seluruh jaringan) untuk tidak menanggapi permintaan ping karena *server* dapat diserang oleh sesuatu yang disebut serangan *denial of service*. Dalam serangan semacam ini, *server* dibanjiri oleh sejumlah besar paket jaringan. Dengan mengabaikan permintaan ping, *server* menjadi kurang mudah diserang.

Akibatnya, perintah ping mungkin berguna untuk memeriksa ketersediaan mesin lokal, tetapi tidak selalu berguna untuk mesin di luar jaringan Anda sendiri.

Perlu diingat

Banyak administrator menggunakan perintah ping dengan nama host, dan jika gagal, gunakan alamat IP untuk melihat apakah kesalahan ada dalam menyelesaikan nama host perangkat. Menggunakan nama host menghemat waktu; jika perintah ping itu berhasil, ada resolusi nama yang sesuai, maka alamat IP juga berfungsi dengan benar.

PERINTAH netstat

Perintah netstat adalah *tools* yang ampuh yang menyediakan informasi jaringan dalam jumlah besar. Perintah netstat dapat digunakan untuk menampilkan informasi tentang koneksi jaringan serta menampilkan tabel routing yang mirip dengan perintah route.

Misalnya, untuk menampilkan informasi statistik mengenai lalu lintas jaringan, gunakan opsi -i pada perintah netstat, seperti berikut:

```
root@localhost:~# netstat -i
Kernel Interface table
Iface   MTU Met   RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0       1500 0       137      0      4 0         12      0      0      0 BMRU
lo        65536 0        18      0      0 0         18      0      0      0 LRU
```

Informasi statistik terpenting dari *output* di atas adalah TX-OK dan TX-ERR. Persentase TX-ERR yang tinggi mungkin menunjukkan adanya masalah pada jaringan, seperti lalu lintas jaringan yang terlalu banyak.

Untuk menggunakan perintah netstat untuk menampilkan informasi routing, gunakan opsi -r:

```
root@localhost:~# netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
192.168.1.0     *               255.255.255.0   U         0 0          0 eth0
default         192.168.1.1     0.0.0.0         UG        0 0          0 eth0
```

Perintah netstat juga dapat digunakan untuk menampilkan port terbuka. Port adalah nomor unik yang dikaitkan dengan layanan yang disediakan oleh *host*. Jika port terbuka, maka layanan atau *service* tersedia bagi *host* lain. Misalnya, Anda dapat masuk ke suatu *host* dari *host* lain menggunakan layanan yang disebut SSH. Layanan SSH diberi port # 22. Jadi, jika port # 22 terbuka, maka layanan tersedia untuk *host* lain.

Penting untuk diperhatikan bahwa *host* juga harus menjalankan layanannya sendiri; ini berarti bahwa layanan (dalam hal ini ssh daemon) yang memungkinkan pengguna jarak jauh untuk masuk harus dimulai (yang biasanya demikian, untuk sebagian besar distro Linux).

Untuk melihat daftar semua port yang saat ini terbuka, gunakan perintah berikut:

```
root@localhost:~# netstat -tln
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 192.168.1.2:53          0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:53            0.0.0.0:*               LISTEN
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp        0      0 127.0.0.1:953           0.0.0.0:*               LISTEN
tcp6       0      0 :::53                   :::*                    LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
tcp6       0      0 ::1:953                 :::*                    LISTEN
```

Seperti yang Anda lihat dari *output* di atas, port # 22 sedang “listening”, yang artinya port tersebut terbuka.

Pada contoh sebelumnya, -t adalah singkatan dari TCP (ingat protokol ini dari awal bab ini), -l adalah singkatan dari listening (port mana yang mendengarkan) dan -n adalah singkatan dari *show number*, bukan nama.

Terkadang menampilkan nama bisa lebih berguna. Ini dapat dilakukan dengan menghapus opsi -n, seperti contoh berikut:

```
root@localhost:~# netstat -tl
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 cserver.example.:domain *:*                     LISTEN
tcp        0      0 localhost:domain        *:*                     LISTEN
tcp        0      0 *:ssh                   *:*                     LISTEN
tcp        0      0 localhost:953           *:*                     LISTEN
tcp6       0      0 [::]:domain             [::]:*                  LISTEN
tcp6       0      0 [::]:ssh                [::]:*                  LISTEN
tcp6       0      0 localhost:953           [::]:*                  LISTEN
```

Pada beberapa distro Linux, Anda mungkin melihat pesan berikut di halaman *manual* perintah netstat:

```
NOTE
     This program is obsolete. Replacement for netstat is ss. Replacement for
     netstat -r is ip route. Replacement for netstat -i is ip -s link.
     Replacement for netstat -g is ip maddr.
```

Meskipun tidak ada pengembangan lebih lanjut yang dilakukan pada perintah netstat, perintah ini masih menjadi *tools* yang sangat baik untuk menampilkan informasi jaringan. Tujuannya adalah untuk mengganti perintah netstat dengan perintah seperti perintah ss dan ip. Namun, penting untuk disadari bahwa ini mungkin membutuhkan waktu.

PERINTAH ss

Perintah ss dirancang untuk menunjukkan statistik soket dan mendukung semua paket utama dan jenis soket. Dimaksudkan sebagai pengganti dan fungsinya mirip dengan perintah netstat, perintah ss juga menunjukkan lebih banyak informasi dan memiliki lebih banyak fitur.

Alasan utama pengguna menggunakan perintah ss adalah untuk melihat koneksi apa yang saat ini dibuat antara mesin lokal dan mesin jarak jauh, statistik tentang koneksi tersebut, dll.

Mirip dengan perintah netstat, Anda bisa mendapatkan banyak informasi berguna dari perintah ss dengan hanya mengetikkan “ss” seperti yang ditunjukkan pada contoh di bawah ini.

```
root@localhost:~# ss
Netid  State      Recv-Q Send-Q         Local Address:Port                 Peer Address:Port
u_str  ESTAB      0      0                     * 104741                    * 104740
u_str  ESTAB      0      0      /var/run/dbus/system_bus_socket 14623      * 14606
u_str  ESTAB      0      0      /var/run/dbus/system_bus_socket 13582      * 13581
u_str  ESTAB      0      0      /var/run/dbus/system_bus_socket 16243      * 16242
u_str  ESTAB      0      0                     * 16009                     * 16010
u_str  ESTAB      0      0      /var/run/dbus/system_bus_socket 10910      * 10909
u_str  ESTAB      0      0      @/tmp/dbus-LoJW0hGFkV 15706                * 15705
u_str  ESTAB      0      0                     * 24997                     * 24998
u_str  ESTAB      0      0                     * 16242                     * 16243
u_str  ESTAB      0      0              @/tmp/dbus-opsTQoGE 15471          * 15470
```

*Output*nya sangat mirip dengan *output* dari perintah netstat tanpa opsi. Kolom di atas adalah:

Netid	Jenis soket dan protokol pengiriman
State	Terhubung atau tidak terhubung, tergantung dari protokol
Recv-Q	Jumlah data yang antri untuk diproses telah diterima
Send-Q	Jumlah data yang antri untuk dikirim ke host lain
Local Address	Alamat dan port pada koneksi host lokal
Peer Address	Alamat dan port pada koneksi host remote

Format *output* dari perintah ss dapat berubah secara drastis, mengingat opsi yang ditentukan, seperti penggunaan opsi -s, yang menampilkan sebagian besar jenis soket, informasi statistik tentang keberadaannya, dan jumlah paket aktual yang dikirim dan diterima melalui masing-masing. jenis soket, seperti yang ditunjukkan di bawah ini:

```
root@localhost:~# ss -s
Total: 1000 (kernel 0)
TCP:   7 (estab 0, closed 0, orphaned 0, synrecv 0, timewait 0/0), ports 0

Transport Total     IP        IPv6
*         0         -         -
RAW       0         0         0
UDP       9         6         3
TCP       7         3         4
INET      16        9         7
FRAG      0         0         0
```

Perlu diingat

Perintah ss biasanya menampilkan banyak baris data, dan mungkin agak sulit untuk mencoba menemukan apa yang Anda inginkan di semua output itu. Pertimbangkan untuk mengirim keluaran ke perintah less agar output lebih mudah dikelola. Pager memungkinkan pengguna untuk menggulir ke atas dan ke bawah, melakukan pencarian dan banyak fungsi berguna lainnya di dalam parameter dari perintah less.

Meskipun perintah ss menawarkan banyak opsi berbeda untuk mengumpulkan dan menampilkan informasi, contoh di atas adalah yang paling umum, dan opsi yang lainnya berada di luar cakupan materi pada level ini.

PERINTAH dig

Mungkin ada kalanya Anda perlu menguji fungsionalitas server DNS yang digunakan oleh *host* Anda. Salah satu cara untuk melakukan ini adalah dengan menggunakan perintah dig, yang melakukan kueri di server DNS untuk menentukan apakah informasi yang diperlukan tersedia di server.

Dalam contoh berikut, perintah dig digunakan untuk menentukan alamat IP dari host example.com:

```
root@localhost:~# dig example.com
; <<>> DiG 9.8.1-P1 <<>> example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 45155
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 0

;; QUESTION SECTION:
;example.com.                   IN      A

;; ANSWER SECTION:
example.com.            86400   IN      A       192.168.1.2
;; AUTHORITY SECTION:
example.com.            86400   IN      NS      example.com.

;; Query time: 0 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Tue Dec  8 17:54:41 2015
;; MSG SIZE  rcvd: 59
```

Perhatikan bahwa respons tersebut menyertakan alamat IP 192.168.1.2, yang berarti bahwa server DNS memiliki alamat IP untuk informasi terjemahan nama host dalam database-nya.

Jika server DNS tidak memiliki informasi yang diminta, maka akan dikonfigurasi untuk meminta *server* DNS lain. Jika tidak ada server DNS yang memiliki informasi yang diminta, maka pesan kesalahan akan muncul:

```
root@localhost:~# dig sample.com
; <<>> DiG 9.8.1-P1 <<>> sample.com
;; global options: +cmd
;; connection timed out; no servers could be reached
```

PERINTAH host

Dalam bentuk yang paling sederhana, perintah *host* bekerja dengan DNS untuk mengasosiasikan nama *host* dengan alamat IP. Seperti yang digunakan dalam contoh sebelumnya, example.com dikaitkan dengan alamat IP 192.168.1.2:

```
root@localhost:~# host example.com
example.com has address 192.168.1.2
```

Perintah *host* juga dapat digunakan secara terbalik jika alamat IP diketahui, tetapi nama domainnya tidak.

```
root@localhost:~# host 192.168.1.2
2.1.168.192.in-addr.arpa domain name pointer example.com.
2.1.168.192.in-addr.arpa domain name pointer cserver.example.com.
```

Ada opsi lain untuk menanyakan berbagai aspek mengenai DNS seperti nama kanonik (Canonical name) CNAME atau alias:

```
root@localhost:~# host -t CNAME example.com
example.com has no CNAME record
```

Karena banyak server DNS yang menyimpan salinan example.com, catatan Start of Authority SOA menunjukkan server utama untuk domain:

```
root@localhost:~# host -t SOA example.com

example.com has SOA record example.com. cserver.example.com. 2 604800 864
00 2419200 604800
```

Daftar lengkap informasi DNS mengenai example.com dapat ditemukan menggunakan opsi -a (all):

```
root@localhost:~# host -a example.com

Trying "example.com"

;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 3549

;; flags: qr aa rd ra; QUERY: 1, ANSWER: 3, AUTHORITY: 0, ADDITIONAL: 1


;; QUESTION SECTION:

;example.com.                   IN      ANY


;; ANSWER SECTION:

example.com.            86400   IN      SOA     example.com. cserver.exam
ple.com. 2 604800 86400 2419200 604800

example.com.            86400   IN      NS      example.com.

example.com.            86400   IN      A       192.168.1.2


;; ADDITIONAL SECTION:

example.com.            86400   IN      A       192.168.1.2


Received 119 bytes from 127.0.0.1#53 in 0 ms
```

PERINTAH ssh

Perintah ssh memungkinkan Anda untuk terhubung ke mesin lain di seluruh jaringan, masuk dan kemudian melakukan tugas pada mesin jarak jauh.

Jika Anda hanya memberikan nama mesin atau alamat IP untuk masuk, perintah ssh mengasumsikan Anda ingin *log in* menggunakan *username* yang sama ketika Anda *log in*. Untuk menggunakan *username* yang berbeda, gunakan sintaks:

```
username@hostname
```

```
root@localhost:~# ssh bob@test
The authenticity of host 'test (127.0.0.1)' can't be established.
RSA key fingerprint is c2:0d:ff:27:4c:f8:69:a9:c6:3e:13:da:2f:47:e4:c9.
Are you sure you want to continue connection (yes/no)? yes
Warning: Permanently added 'test' (RSA) to the list of known hosts.
bob@test's password:
bob@test:~$ date
Fri Oct   4 16:14:43 CDT 2013
```

Untuk kembali ke mesin lokal, gunakan perintah *exit*:

```
bob@test:~$ exit
logout
Connection to test closed.
root@localhost:~#
```

Peringatan: Hati-hati, jika Anda menggunakan perintah *exit* terlalu sering, Anda akan menutup jendela terminal tempat Anda bekerja!

RSA KEY FINGERPRINT

Saat menggunakan perintah ssh, *prompt* pertama meminta Anda untuk memverifikasi identitas mesin yang Anda masuki. Dalam kebanyakan kasus, Anda ingin menjawab *yes*. Meskipun Anda dapat memeriksa dengan administrator mesin jarak jauh untuk memastikan bahwa kunci RSA *fingerprint* sudah benar, ini bukanlah tujuan dari kueri ini. Ini dirancang untuk upaya masuk di masa mendatang.

Setelah Anda menjawab *yes*, sidik jari kunci RSA dari mesin jarak jauh disimpan di sistem lokal Anda. Saat Anda mencoba melakukan ssh ke mesin yang sama di masa mendatang, sidik jari kunci RSA yang disediakan oleh mesin jarak jauh dibandingkan dengan salinan yang disimpan di mesin lokal. Jika cocok, maka nama pengguna akan muncul. Jika tidak cocok, kesalahan seperti berikut akan ditampilkan:

```
sysadmin@localhost:~$ ssh bob@test
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!    @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that the RSA host key has just been changed.
The fingerprint for the RSA key sent by the remote host is
c2:0d:ff:27:4c:f8:69:a9:c6:3e:13:da:2f:47:e4:c9.
Please contact your system administrator.
Add correct host key in /home/sysadmin/.ssh/known_hosts to get rid of this message.
Offending key in /home/sysadmin/.ssh/known_hosts:1
RSA host key for test has changed and you have requested strict checking.
Host key verification failed.
```

Kesalahan ini dapat menunjukkan bahwa *host* penipu telah mengganti *host* yang benar. Tanyakan kepada administrator sistem jarak jauh. Jika sistem baru saja diinstal ulang, itu akan memiliki kunci RSA baru, dan itu akan menyebabkan kesalahan ini.

Jika pesan kesalahan ini disebabkan oleh instalasi ulang mesin jarak jauh, Anda dapat menghapus file ~/.ssh/known_hosts dari sistem lokal Anda (atau cukup hapus entri untuk satu mesin itu) dan coba sambungkan lagi:

```
sysadmin@localhost:~$ cat ~/.ssh/known_hosts

test ssh-rsa AAAAB3NzaC1yc2EAAAAmIwAAAQEAklOUpkDHrfHY17SbrmTIp/RZ0V4DTxgq
9wzd+ohy006SWDSGPA+nafzlHDPOW7vdI4mZ5ew18KL4JW9jbhUFrviQzM7xlELEVf4h9lFX5
QVkbPppSrg0cda3Pbv7kOdJ/MTyBlWXFCRH+Cv3FXRitBqxiX1nKhXpHAZsMciLq8V6RjsNAQ
wdsdMFvSlVK/7BA

t5FaiKoAfncM1Q8x3+2V0Ww71/eIFmb1zuUFljHYTprrX88XypNDvjYNby6vw/Pb0rwprz/Tn

mZAW3UX+PnTPI89ZPmNBLuxyrD2cE86Z/il8b+gw3r3+1nJotmIkjn2so1d01QraTlMqVSsbx

NrRFi9wrf+ghw==

sysadmin@localhost:~$ rm ~/.ssh/known_hosts

sysadmin@localhost:~$ ssh bob@test

The authenticity of host 'test (127.0.0.1)' can't be established.

RSA key fingerprint is c2:0d:ff:27:4c:f8:69:a9:c6:3e:13:da:2f:47:e4:c9.

Are you sure you want to continue connection (yes/no)? yes

Warning: Permanently added 'test' (RSA) to the list of known hosts.

bob@test's password:
```

RANGKUMAN

1. Istilah-istilah yang berkaitan dengan jaringan antara lain adalah *host*, *network*, internet, Wi-Fi, *server*, *service*, *client*, dan router.
2. Istilah pada jaringan yang lebih fokus pada berbagai layanan jaringan antara lain: network packet, IP address, netmask, hostname, URL, DHCP, DNS, Ethernet, dan TCP/IP.
3. File yang digunakan untuk name resolution pada host Linux adalah file /etc/hosts, /etc/resolv.conf, dan /etc/nsswitch.conf.
4. Perintah-perintah yang digunakan untuk gunakan untuk melihat informasi jaringan serta untuk troubleshoot jaringan pada Linux antara lain ada perintah ifconfig, ip, route, ping, netstat, ss, dig, host, dan ssh.
5. Perintah ifconfig dan ip digunakan untuk menampilkan informasi konfigurasi jaringan.
6. Perintah route digunakan untuk menampilkan tabel routing.
7. Perintah ping digunakan untuk mengelola status konektivitas jaringan antara source dan perangkat melalui jaringan IP. Melihat apakah sebuah komputer terhubung dengan komputer lainnya.
8. Perintah netstat berguna untuk menampilkan informasi tentang lalu lintas transfer data serta menampilkan tabel routing.
9. Perintah ss digunakan untuk menampilkan informasi statistik soket, mendukung semua jenis paket utama dan socket.
10. Perintah dig digunakan untuk melakukan permintaan pada DNS server dan menentukan apakah informasi yang diperlukan tersedia pada server.
11. Perintah host digunakan untuk pencarian DNS.
12. Perintah ssh digunakan untuk mengakses mesin (*server*) jarak jauh secara remote melalui koneksi yang terenkripsi.